

<b>PJT명</b>	<b>Django 웹 프레임워크의 사용자 인증 기반 DB 설계</b>	
<b>단계</b>	[Django 개발]	
<b>진행일자</b>	2025.10.02	
<b>예상 구현 시간</b>	필수기능	5H
	추가기능	2H
	심화기능	1H

## 1. 목표

- 데이터를 생성, 조회, 수정, 삭제할 수 있는 Web application을 제작한다.
- Django Model과 ORM에 대하여 이해한다.
- Django Form에 대하여 이해한다.
- Django Authentication System에 대하여 이해한다.

## 2. 준비사항

### 1) 제공 파일

- 프로젝트를 진행하는데 활용할 수 있는 이미지가 담긴 assets 폴더가 제공됨

### 2) 개발언어 및 툴

- Python 3.11
- Django 5.2
- HTML, CSS
- Visual Studio Code
- Google Chrome

### 3) 필수 라이브러리 / 오픈소스

- Bootstrap 5.3

### 3. 작업 순서

- 1) 팀원과 같이 요구사항을 확인하고, GitLab에 프로젝트를 생성한다.
  - 프로젝트 이름은 05-pjt로 지정한다.
  - 각 반 담당 강사님을 Maintainer로 설정한다.
- 2) 팀원과 합의하여 협업 방식을 결정하고, 요구사항을 구현한다.
- 3) 작성한 코드들을 정리하고, README를 작성한다.
  - .gitignore 파일을 활용하여 불필요한 파일 및 폴더는 제출하지 않는다.
- 4) README 작성이 완료되면 심화 학습을 진행한다.
- 5) 제출 기한에 맞춰 모든 산출물이 GitLab에 업로드 될 수 있도록 한다.

## 4. 요구사항

AI 기반 도서 분석과 창작 지원 통합 솔루션 서비스를 구축하려고 한다. 다양한 장르와 형태의 도서 데이터를 수집 및 관리하고, 이를 기반으로 AI 기반 도서 분석 기능을 설계 및 구현한다. 또한 작가들에게 아이디어 발상 지원, 플롯 구성 제안, 문장 교정 및 스타일 개선 등 창작 전반에 걸친 다양한 AI 기반 창작 지원 기능을 제공한다. 또한 도서에 대한 사용자 리뷰 및 감상평 공유 커뮤니티 기능을 제공하여, 사용자들이 활발하게 소통하고 정보를 교환할 수 있는 기능을 제공한다. 사용자는 자신이 읽은 도서를 평가하고, 다른 사용자의 리뷰를 참고하여 다음 도서를 선택하는데 도움을 받을 수 있다. 나아가, 관심 도서 목록을 맞춤형으로 구성하는 등 다양한 편의 기능을 제공한다. 팀원과 상의하여 아래 요구사항을 만족할 수 있도록 요구 사항 명세서를 작성 및 구현해보자.

서비스의 인증 시스템을 구성하는 단계이다. 도서 데이터의 생성, 조회, 수정, 삭제가 가능하며, 로그인, 로그아웃, 회원가입, 회원탈퇴, 회원정보수정, 비밀번호 변경이 가능한 애플리케이션을 완성해보자.

- 요구사항 예시(참고용)
  - 아래의 내용을 참고하여 추가적인 아이디어에 대해 요구사항을 추가 또는 수정하여 기능을 구현한다. 단, **필수 기능은 구현해야 하며, 수정할 수 없다.**

번호	분류	요구사항명	요구사항 상세	우선순위
기능적 요구사항				
F01	프로젝트	프로젝트 구성	도서 커뮤니티 서비스 구현을 위한 Django 프로젝트 및 앱 생성	필수
F02	accounts	Model Class	사용자 데이터를 데이터베이스에 저장할 수 있도록 AbstractUser를 상속받는 Django Model 클래스 구현	필수
F03	accounts	Form Class	사용자가 입력한 사용자 데이터 검증을 위해 적절한 Form 클래스 활용 및 구현	필수
F04	books	Model Class	도서 데이터를 데이터베이스에 저장할 수 있도록 Django Model 클래스 구현	필수
F05	books	Form Class	사용자가 입력한 도서 데이터 검증을 위한 Django Form 클래스 구현	필수
F06	accounts view 함수	login	로그인 UI를 출력하고, 유효성 검증 및 세션 데이터를 저장하는 view 함수 구현	필수
F07	accounts view 함수	logout	현재 인증된 사용자의 세션 관련 데이터를 삭제하는 함수 구현	필수
F08	accounts view 함수	signup	회원가입 UI를 출력하고, 유효성 검증 및 회원 데이터를 저장하는 view 함수 구현	필수
F09	accounts view 함수	update	회원정보수정 UI를 출력하고, 유효성 검증 및 회원 데이터를 수정하는 view 함수 구현	필수
F10	accounts view 함수	delete	현재 인증된 사용자의 데이터를 삭제하는 함수 구현	필수
F11	accounts view 함수	change_password	비밀번호 변경 UI를 출력하고, 유효성 검증 및 비밀번호를 변경하는 view 함수 구현	필수
F12	books view 함수	index	전체 도서 데이터를 조회하는 view 함수 구현	필수
F13	books view 함수	create	도서 데이터를 입력할 수 있는 UI를 제공하고, 데이터에 따라 도서 데이터를 저장하는 view 함수 구현	필수
F14	books view 함수	detail	단일 도서 데이터를 조회하는 view 함수 구현	필수

F15	books view 함수	update	도서 데이터를 입력할 수 있는 UI를 제공하고, 선택한 도서 데이터를 수정하는 view 함수 구현	필수
F16	books view 함수	delete	단일 도서 데이터를 삭제하는 view 함수 구현	필수
F17	AI 활용	정보 생성 및 저장	생성형 AI API를 활용하여 작가 정보 검색 및 생성	도전
F18	AI 활용	User 모델 커스텀	사용자에 대한 추가 정보를 저장 및 관리	도전
F19	AI 활용	프로필 페이지	개별 사용자의 정보를 확인할 수 있는 페이지 구현	도전
비기능적 요구사항				
NF01	유지보수	URL 구성	URL이 변경될 때 유지보수가 용이하도록 구현	필수
NF02	보안	HTTP Method 허용	허용된 HTTP Method를 사용하는 요청만 허락하도록 구현	필수

## 1) 프로젝트 및 앱 (필수)

도서 데이터를 생성, 조회, 수정, 삭제할 수 있는 Django 프로젝트를 만든다. 명시된 요구사항 이외 서비스를 위해 필요하다고 생각되는 기능들은 자유롭게 구현해도 무관하며, Bootstrap을 활용하여 자유롭게 스타일링 한다.

- 요구사항 번호: F01, NF01, NF02
- 프로젝트 이름: mypjt
- 앱 이름: books, accounts
- 페이지간 이동 편의성을 갖추기 위한 Navbar 구성
  - books의 index, create와 accounts의 기능들에 접근하기 편하도록 링크를 제공
  - 다양한 페이지에서 도달할 수 있도록 여러 페이지에 동일하게 출력되도록 구현

## A. accounts

사용자 데이터를 관리하기 위한 앱이다.

- 요구사항 번호: F02, F03
- Model 클래스
  - Django에서 제공하는 AbstractUser를 상속받는 커스텀 모델 클래스 구현
- Form 클래스
  - 상황에 따라 Django가 제공하는 Form과 필요한 경우 커스텀 Form 클래스를 적절히 활용
- 각 세부 기능들을 위한 view 함수를 구성
  - login, logout, signup, update, delete, change\_password
  - URL이 변경될 가능성을 고려하여 유지보수에 유리한 형태로 구성
  - 각 view 함수가 허용하는 HTTP Method에만 호출되도록 구현

## B. books

도서 데이터를 관리하기 위한 앱이다.

- 요구사항 번호: F04, F05
- Model 클래스 이름: Book
  - 도서 제목, 설명, 회원 리뷰 평점, 저자를 저장할 필드 4개 지정
- Form 클래스 이름: BookForm
  - Book 모델 데이터 검증, 저장, 에러메시지 등을 관리하기 위해 적절한 ModelForm 사용
- 각 세부 기능들을 위한 view 함수를 구성
  - index, create, detail, update, delete
  - URL이 변경될 가능성을 고려하여 유지보수에 유리한 형태로 구성
  - 각 view 함수가 허용하는 HTTP Method에만 호출되도록 구현

## 2) accounts 앱 view 함수

사용자 데이터를 관리하는 역할을 하는 accounts 앱의 view 함수를, Django Authentication System을 활용하여 구현하자.

### A. login

사용자가 자기 정보를 입력할 수 있는 UI를 제공하고, 로그인할 수 있도록 해주는 login view 함수를 구현하자.

- 요구사항 번호: F06
- 사용자가 로그인 되어 있지 않을 때 정상적으로 동작
  - 로그인 되어있지 않은 상태에서만 출력되는 로그인 링크를 통해 이동
  - 사용자가 제공한 데이터의 유효성 검증이 이뤄져야 함
  - 로그인에 성공할 경우 홈페이지로 리다이렉트
- 사용자의 로그인 여부에 따라 사이트 UI가 변화되어야 함
  - 로그인 링크가 회원정보수정, 회원탈퇴, 로그아웃 링크 등으로 변경
  - 로그인이 되어있지 않은 경우 데이터 생성 링크 출력되지 않음
- 같은 view 함수에서 HTML 제공과 로그인이 같이 이뤄지도록 구현
  - GET 요청을 받을 경우: HTML 제공
  - POST 요청을 받을 경우: 유효한 사용자 데이터의 경우 세션 정보 생성(로그인), 아닌 경우 오류를 페이지에 표시



- 출력 예시

- 로그인 페이지

ThRead

INDEX LOGIN SIGNUP

로그인

Username:

Password:

로그인

- 로그인 전후 링크 변경

INDEX LOGIN SIGNUP

INDEX CREATE MY PROFILE LOGOUT

## B. logout

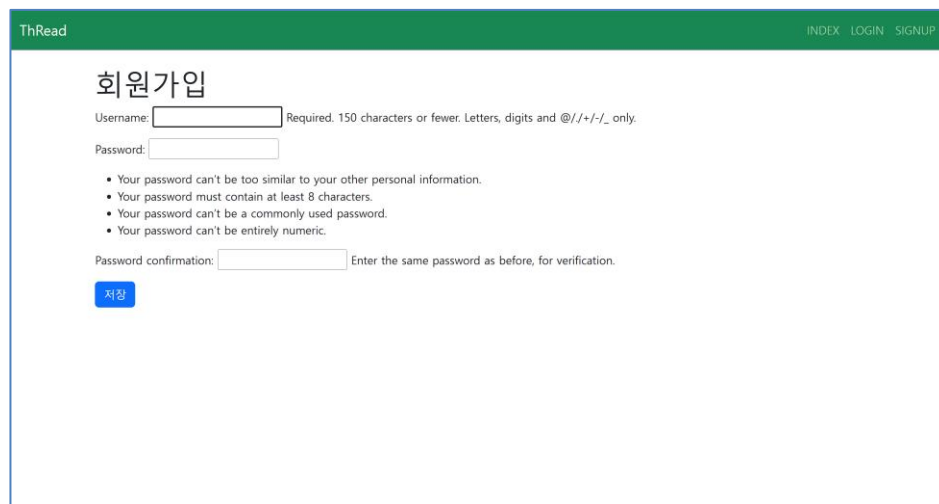
사용자가 로그아웃 할 수 있는 UI를 제공하고, 로그아웃을 해주는  
logout view 함수를 구현하자.

- 요구사항 번호: F07
- 사용자가 로그인 되어 있을 때 정상적으로 동작
  - 로그인 된 상태에서만 출력되는 로그아웃 UI를 통해 사용
  - 로그아웃에 성공할 경우 홈페이지로 리다이렉트
- POST 요청에 대해서만 작동하도록 구현

## C. signup

아직 등록되지 않은 사용자가 데이터를 입력할 수 있는 UI를 제공하고, 회원가입을 진행할 수 있도록 해주는 signup view 함수를 구현하자.

- 요구사항 번호: F08
- 사용자가 로그인 되어 있지 않을 때 정상적으로 동작
  - 로그인 되어있지 않은 상태에서만 출력되는 회원가입 링크를 통해 이동
  - 사용자가 제공한 데이터의 유효성 검증이 이뤄져야 함
  - 회원가입 완료 시 로그인을 진행하고 홈페이지로 리다이렉트
- 같은 view 함수에서 HTML 제공과 회원가입이 같이 이뤄지도록 구현
  - GET 요청을 받을 경우: HTML 제공
  - POST 요청을 받을 경우: 유효한 데이터의 경우 새로운 사용자 정보 생성, 세션 정보 생성(로그인), 아닌 경우 오류를 페이지에 표시
- 출력 예시



ThRead INDEX LOGIN SIGNUP

### 회원가입

Username:  Required. 150 characters or fewer. Letters, digits and @/+/+/-/\_ only.

Password:

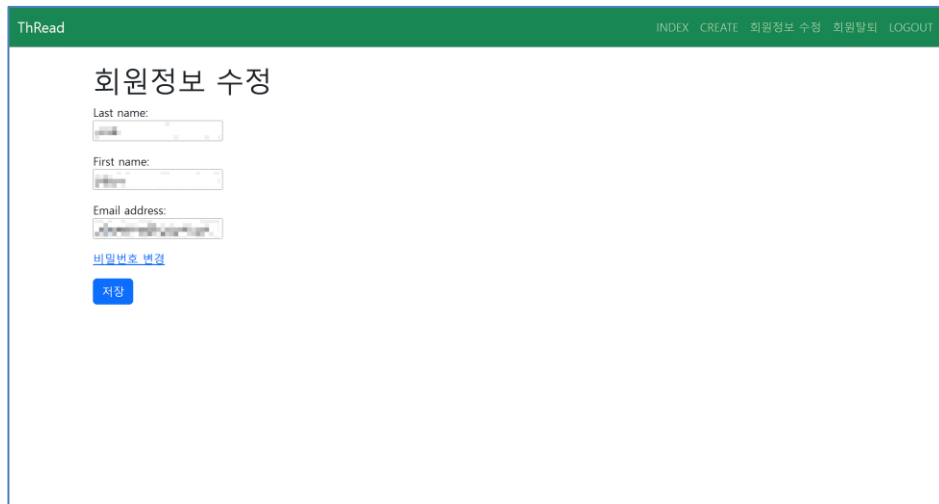
- Your password can't be too similar to your other personal information.
- Your password must contain at least 8 characters.
- Your password can't be a commonly used password.
- Your password can't be entirely numeric.

Password confirmation:  Enter the same password as before, for verification.

## D. update

사용자가 자신의 정보를 입력할 수 있는 UI를 제공하고, 회원 정보를 수정할 수 있도록 해주는 update view 함수를 구현하자.

- 요구사항 번호: F09
- 사용자가 로그인 되어 있을 때 정상적으로 동작
  - 로그인 된 상태에서만 출력되는 회원정보수정 링크를 통해 이동
  - 회원정보수정에 성공할 경우 홈페이지로 리다이렉트
- 같은 view 함수에서 HTML 제공과 회원정보수정이 같이 이뤄지도록 구현
  - GET 요청을 받을 경우: HTML 제공, 단 일반적인 사용자가 접근하면 안 되는 정보는 출력되지 않아야 함
  - POST 요청을 받을 경우: 유효한 데이터의 경우 로그인 된 사용자 정보 수정, 아닌 경우 오류를 페이지에 표시
- 출력 예시



The screenshot shows a web application interface for updating member information. At the top, there is a green navigation bar with the text 'ThRead' on the left and a series of links 'INDEX CREATE 회원정보 수정 회원탈퇴 LOGOUT' on the right. The main content area has a title '회원정보 수정' (Update Member Information). Below the title, there are three input fields: 'Last name:' with a text input, 'First name:' with a text input, and 'Email address:' with a text input. Below these fields, there is a blue link labeled '비밀번호 변경' (Change Password). At the bottom of the form, there is a blue button labeled '저장' (Save).

## E. delete

사용자가 회원탈퇴를 할 수 있는 UI를 제공하고, 회원탈퇴를 해주는 delete view 함수를 구현하자.

- 요구사항 번호: F10
- 사용자가 로그인 되어 있을 때 정상적으로 동작
  - 로그인 된 상태에서만 출력되는 회원탈퇴 UI를 통해 사용
  - 회원탈퇴에 성공할 경우 홈페이지로 리다이렉트
- POST 요청에 대해서만 작동하도록 구현

## F. change\_password

사용자가 자신의 비밀번호를 수정할 수 있도록 비밀번호를 입력하는 UI를 제공하고, 비밀번호를 수정해주는 `change_password` view 함수를 구현하자.

- 요구사항 번호: F11
- 사용자가 로그인 되어 있을 때 정상적으로 동작
  - 로그인 된 상태에서만 출력되는 비밀번호수정 UI, 또는 회원정보수정 페이지에 Django가 기본적으로 제공하는 링크를 통해 이동
  - 비밀번호 수정에 성공할 경우 홈페이지로 리다이렉트
  - 비밀번호 수정 후 강제로 로그아웃 되지 않도록 구현
- 같은 view 함수에서 HTML 제공과 비밀번호변경이 같이 이뤄지도록 구현
  - GET 요청을 받을 경우: HTML 제공
  - POST 요청을 받을 경우: 유효한 데이터의 경우 로그인 된 사용자의 비밀번호 수정, 아닌 경우 오류를 페이지에 표시
- 출력 예시

ThRead INDEX CREATE 회원정보 수정 회원탈퇴 LOGOUT

### 비밀번호 변경

Old password:

New password:

- Your password can't be too similar to your other personal information.
- Your password must contain at least 8 characters.
- Your password can't be a commonly used password.
- Your password can't be entirely numeric.

New password confirmation:  Enter the same password as before, for verification.

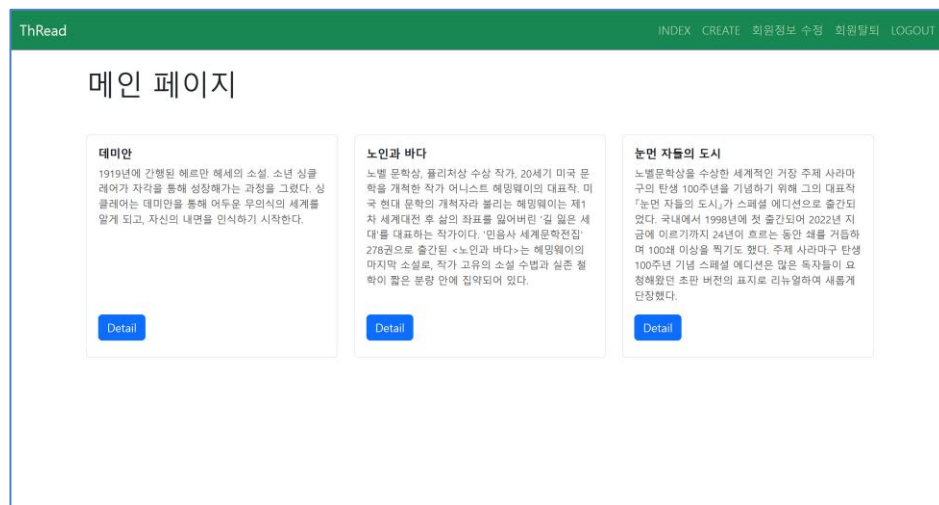
### 3) books 앱 view 함수

도서 데이터를 생성, 조회, 수정, 삭제하는 역할을 하는 books 앱을 구현하자.

#### A. index

전체 도서 데이터를 조회하기 위한 index view 함수를 구현한다.

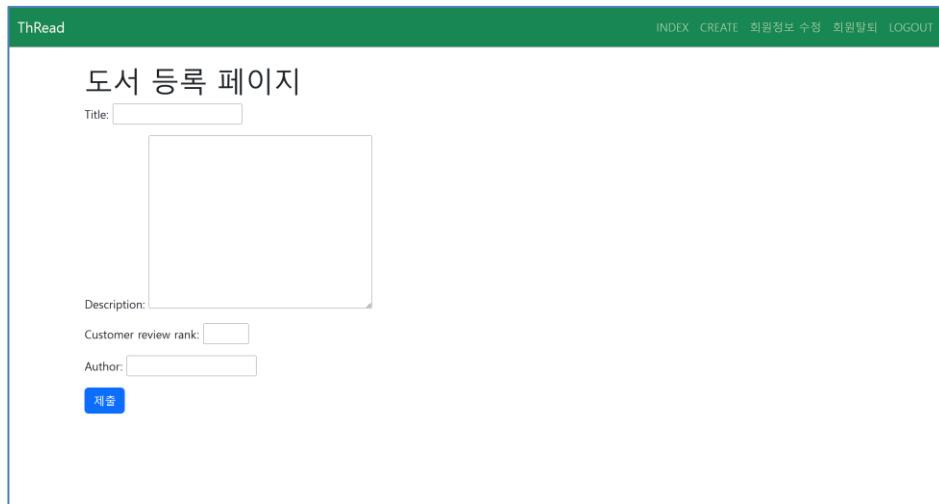
- 요구사항 번호: F12
- 전체 도서 데이터가 조회되어야 함
- 특정 도서를 선택하여 상세 페이지로 이동할 수 있는 링크가 제공되어야 함
  - 링크의 문구와 UI는 자유롭게 구성
- 출력 예시



## B. create

도서 데이터를 입력하기 위한 UI를 제공하고, 입력된 데이터를 저장하는 create view 함수를 구현한다.

- 요구사항 번호: F13
- Book 모델의 데이터를 입력할 수 있는 UI 구현
  - 유효하지 않은 데이터가 저장되지 않도록 구현
  - 유효한 데이터의 경우 저장 후 상세 페이지로 리다이렉트
  - UI 디자인은 자유롭게 구성
- 같은 view 함수에서 HTML 제공과 데이터 생성이 같이 이뤄지도록 구현
  - GET 요청을 받을 경우: HTML 제공
  - POST 요청을 받을 경우: 형식에 맞는 데이터는 저장, 형식이 맞지 않는 데이터는 오류를 페이지에 표시
- 출력 예시

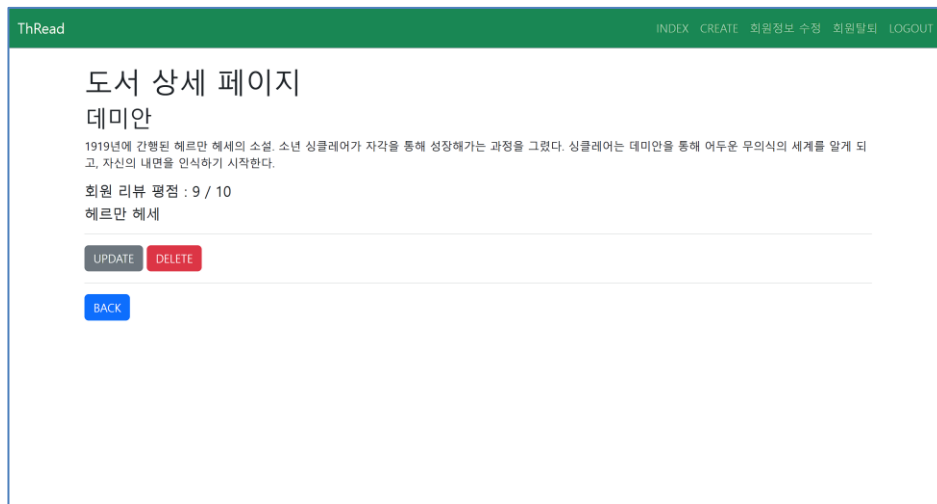




## C. detail

단일 도서 데이터를 조회하는 detail view 함수를 구현한다.

- 요구사항 번호: F14
- index 페이지에서 도서를 선택했을 때 이동됨
  - 선택한 해당 도서의 정보를 바탕으로 내용이 출력되어야 함
- 도서 제목, 설명, 회원 리뷰 평점, 저자 데이터를 출력
  - 도서를 수정하기 위한 페이지로 이동하는 링크 구현
  - 도서를 삭제하는 버튼 구현
  - 이전 페이지(index)로 이동하는 링크 구현
  - UI 디자인은 자유롭게 구성
- 출력 예시



## D. update

도서 데이터를 수정하기 위한 UI를 제공하고, 입력된 데이터를 바탕으로 대상 도서 데이터를 수정하는 update view 함수를 구현한다.

- 요구사항 번호: F15
- detail 페이지에서 수정하기 링크를 통해 이동됨
  - detail 페이지에서 제공되었던 도서 정보를 바탕으로 기능이 동작되도록 구현
- Book 모델의 데이터를 입력할 수 있는 UI 구현
  - 이전 도서의 데이터가 미리 입력되어 있도록 구현
  - 유효하지 않은 데이터가 저장되지 않도록 구현
  - 유효한 데이터의 경우 저장 후 상세 페이지로 리다이렉트
  - UI 디자인은 자유롭게 구성
- 같은 view 함수에서 HTML 제공과 데이터 수정이 같이 이뤄지도록 구현
  - GET 요청을 받을 경우: HTML 제공
  - POST 요청을 받을 경우: 형식에 맞는 데이터는 저장, 형식이 맞지 않는 데이터는 오류를 페이지에 표시

### ● 출력 예시

ThRead INDEX CREATE 회원정보 수정 회원탈퇴 LOGOUT

### 도서 수정 페이지

Title: 데미안

Description: 1919년에 간행된 헤르만 헤세의 소설. 소년 싱클레어가 자각을 통해 성장해가는 과정을 그렸다. 싱클레어는 데미안을 통해 어두운 무의식의 세계를 알게 되고, 자신의 내면을 인식하기 시작한다.

Customer review rank: 9

Author: 헤르만 헤세

제출

## E. delete

도서 데이터를 삭제하는 기능을 제공하는 delete view 함수를 구현한다.

- 요구사항 번호: F16
- detail 페이지에서 삭제하기 버튼을 통해 호출됨
  - detail 페이지에서 제공되었던 도서 정보를 바탕으로 기능이 동작하도록 구현
  - POST 요청에 대해서만 작동하도록 구현
- 삭제가 된 이후에는 홈페이지로 리다이렉트

#### 4) 도전 과제

생성형 AI 도구를 활용하여 도전과제 요구사항을 해결해보자.

- 코드 생성, 아이디어 구상, 문제 해결 방법 탐색 등 다양한 방식으로 활용할 수 있다.
- 사용할 생성형 AI 서비스는 자유롭게 선택한다.
  - 제공되는 GPT API Key를 활용할 경우, 모델은 반드시 gpt-5-nano 모델을 사용한다.
- 최종 결과물은 AI 생성 내용을 바탕으로 직접 수정 및 개선하여 적용해본다.

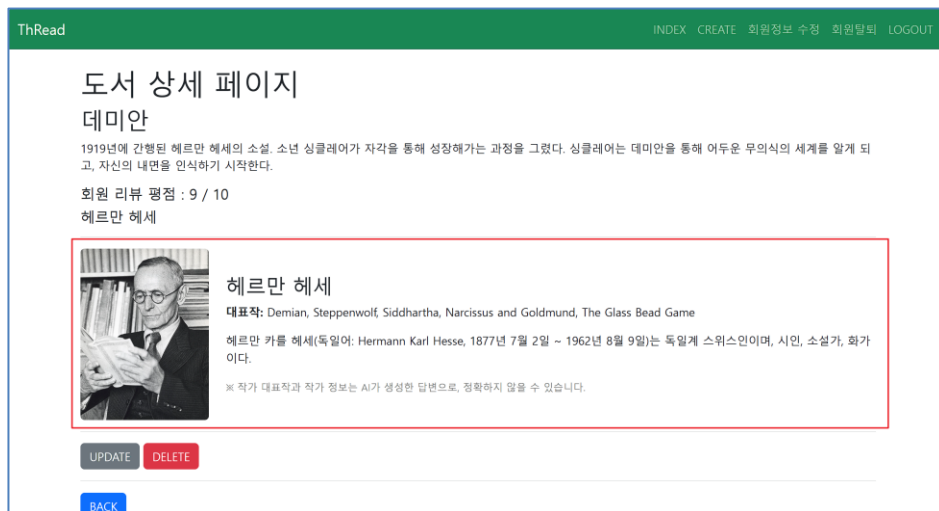
보조 수단으로 활용하되, 능동적인 자세로 학습에 임할 것.

- 최종적인 이해와 적용은 자기 주도적 학습을 통해 이루어지며, 배운 내용을 스스로 기록하고 정리하며 학습 효과를 높일 것.

## A. 생성형 AI API를 활용한 작가 정보 생성 및 저장

생성형 AI API를 활용하여 작가 정보를 생성하고 저장해보자.

- 요구사항 번호: F17
- 도서를 등록할 때 BookForm으로 입력받은 작가 이름을 위키피디아 API의 쿼리로 전달해 작가 정보, 프로필 사진 검색
- 사용자 입력 데이터와 위키피디아 검색 정보를 바탕으로, GPT API를 활용하기
  - 작가 정보와 대표 작품 목록 요청
  - JSON 형태로 응답하도록 프롬프트 작성
- Book 모델에 필드를 추가하여 저장
  - author\_info와 author\_works 필드에 추가
- 추가한 필드를 도서 상세 페이지에 출력
  - 작가 정보 영역 아래에 "작가 대표작과 작가 정보는 AI가 생성한 정보로, 정확하지 않을 수 있습니다."라는 안내 문구 추가
- 출력 예시



## B. User 모델 커스텀

사용자가 회원가입 시 추가 정보를 입력할 수 있도록 User 모델을 커스텀해보자.

- 요구사항 번호: F18
- User 모델에 추가 정보를 위한 필드를 추가
  - 성별, 나이, 주간 평균 독서 시간, 연간 독서량, 프로필 사진, 관심 장르
  - 나이, 주간 평균 독서 시간, 연간 독서량은 양의 정수만 입력할 수 있도록 설정
  - 관심 장르 필드는 CharField, 최대 길이 600으로 설정
  - 업로드 된 프로필 사진은 media/profile\_images에서 관리
- 회원가입 Form: CustomUserCreationForm
  - UserCreationForm을 상속받아 구현
  - MultipleChoiceField, CheckboxSelectMultiple로 interested\_genres 데이터를 복수로 선택 가능하게 구현
  - Meta를 활용하여 모든 필드를 추가하고, 한글로 label을 지정
  - 생성자를 활용하여 password1, password2의 label을 한글로 재설정
- 회원정보 수정 Form: CustomUserChangeForm
  - UserChangeForm을 상속받아 구현
  - 회원가입 Form과 동일하게 interested\_genres 구현
  - 회원가입 Form과 동일하게 Meta에 필요한 필드 정의
- 회원가입 view: signup view 함수
  - CustomUserCreationForm을 이용하여 유효성 검사
  - 유효한 Form의 경우 User 객체를 생성하고, cleaned\_data에서 관심 장르 추출
  - 관심 장르를 각각 "," 로 구분하여 연결하고(','.join()), User 객체의 interested\_genres 필드에 할당

- 이후 User 저장 후 홈페이지로 리다이렉트
- 회원정보 수정 view: update view 함수
  - CustomUserChangeForm을 이용하여 유효성 검사
  - GET 요청 시 User 객체의 interested\_genres 문자열 확인, 쉼표로 분리하여 리스트로 변환 후 Form의 initial 값으로 활용
  - POST 요청 시 interested\_genres를 cleaned\_data에서 추출 후 각각 "," 로 구분하여 연결하고(','.join()), User 객체의 interested\_genres 필드에 할당
  - 데이터 저장 후 홈페이지로 리다이렉트

- 출력 예시

- 회원가입 페이지 a

ThRead

INDEX LOGIN SIGNUP

회원가입

아이디:

비밀번호:

비밀번호 확인:

이메일:

성:

이름:

성별:  

-----v

나이:

주간 평균 독서 시간(시간):

연간 독서량(권):

관심 장르:  

☐ 소설/시/희곡

☐ 경제/경영

☐ 자기계발

☐ 인문/교양

☐ 과학

☐ 취미/실용

☐ 어린이/청소년

저장



- 회원 정보 수정 페이지

ThRead

INDEXCREATEMY PROFILELOGOUT

### 회원정보 수정

성:

이름:

이메일:

성별:   
남성 ☐

나이:

40

주간 평균 독서 시간(시간):

1

연간 독서량(권):

12

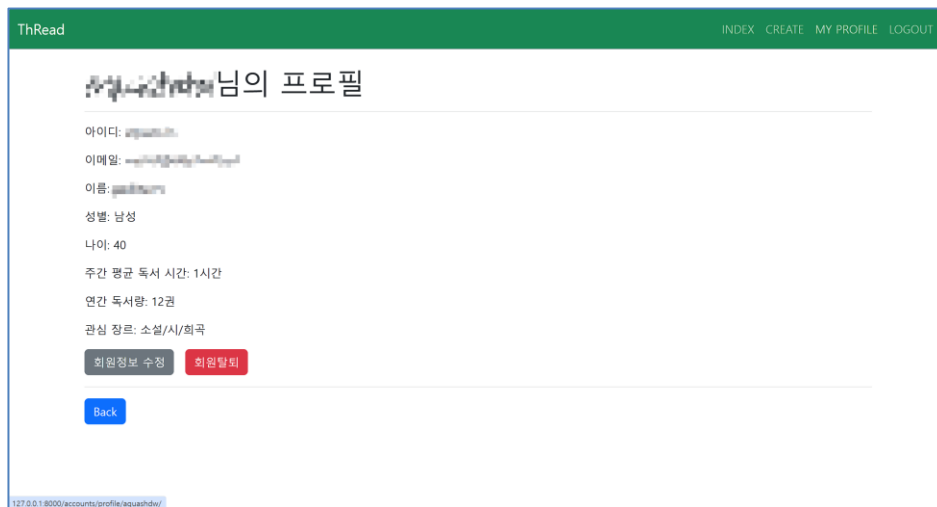
관심 장르:   
☒ 소설/시/희곡   
☐ 경제/경영   
☐ 자기계발   
☐ 인문/교양   
☐ 과학   
☐ 취미/실용   
☐ 어린이/청소년

[비밀번호 변경](#)

## C. 사용자별 개인 프로필 페이지

User 모델의 정보를 활용하여 사용자 개인 프로필 페이지를 만들어보자.

- 요구사항 번호: F19
- 다음 정보가 출력되도록 구현
  - 프로필 사진
  - 아이디, 이메일, 이름 (성 + 이름)
  - 성별, 나이, 주간 평균 독서 시간, 연간 독서량, 관심 장르
- 회원 정보 수정, 회원 탈퇴 링크를, 내 프로필로 이동하는 링크로 대체
  - 대체된 링크는 내 프로필 페이지에서 사용할 수 있도록 구현
- 출력 예시



## 5. 참고자료

- Django 5.2  
<https://www.djangoproject.com/start/overview/>
- Python OpenAI Library  
<https://platform.openai.com/docs/libraries?language=python>  
<https://github.com/openai/openai-python>

## 6. 결과

제출 기한은 진행일 18시까지이므로 제출 기한을 지킬 수 있도록 한다. 제출은 GitLab을 통해서 이뤄진다.

- 산출물과 제출
  - 단계별로 구현 과정 중 학습한 내용, 어려웠던 부분, 새로 배운 것들 및 느낀 점을 상세히 기록한 README.md
  - 완성된 각 문제 별 소스코드 및 실행 화면 캡처본
  - 프로젝트 이름은 05-pjt로 지정, 각자의 계정에 생성할 것
  - 각 반 담당 강사님을 Maintainer로 설정

- 끝 -